Case-based maintenance : Structuring and incrementing the Case.

Brigitte Morello, Mohamed Karim Haouchine, Noureddine Zerhouni

Case-based Maintenance: Structuring and incrementing the Case Base

Brigitte Chebel-Morello[1], Mohamed Karim Haouchine[2], Noureddine Zerhouni[12]

[1] FEMTO-ST institute, Besançon, France

[2] ENSMM, Besançon, France

(brigitte.morello@femto-st.fr ; noureddine.zerhouni@femto-st.fr)

Abstract.

To avoid performance degradation and maintain the quality of results obtained by the case-based reasoning (CBR) systems, maintenance becomes necessary, especially for those systems designed to operate over long periods and which must handle large numbers of cases. CBR systems cannot be preserved without scanning the case base. For this reason, the latter must undergo maintenance operations.
The techniques of case base's dimension optimization is the analog of instance reduction size methodology (in the machine learning community). This study links these techniques by presenting case-based maintenance in the framework of instance based reduction, and provides: first an overview of CBM studies, second, a novel method of structuring and updating the case base and finally an application of industrial case is presented.

The structuring combines a categorization algorithm with a measure of competence CM based on competence and performance criteria. Since the case base must progress over time through the addition of new cases, an auto-increment algorithm is installed in order to dynamically ensure the structuring and the quality of a case base. The proposed method was evaluated through a case base from an industrial plant. In addition, an experimental study of the competence and the performance was undertaken on reference benchmarks. This study showed that the proposed method gives better results than the best methods currently found in the literature.

Keywords: Case-based mining, case-based maintenance, prototyping, instance reduction, auto-increment, competence, performance.

1. INTRODUCTION

Case-Based Reasoning (CBR) is an approach to problem solving and learning through the storing of solutions to similar problems such as cases in a memory called a case base (Aamodt and Plaza, 1994). The case is a body of knowledge representing an experience, defined to a description of the problem resolution event (new case). It is composed of two options: a description of the situation representing a problem and a solution used to remedy this situation (case = (Problem P, Solution S)). A case is placed in a case base and is called a source case which will be used to solve a new case called the target case. A general CBR cycle may be described as having five phases: elaborating, retrieving, reusing, revising and retaining. From a case to be solved, the elaboration phase builds a new problem in a target case by completing or altering the problem description from a possibly incomplete description. According to a similarity metric, cases similar to the target case are first found (the retrieving step) and then adapted by solution construction (reusing). Finally the solution is

validated if necessary. The retaining phase consists of storing the new case once validated, provided storage is considered relevant.

A CBR System contains four knowledge containers (Richter, 1998): 1) the vocabulary knowledge describing the case and the problem domain, 2) the retrieval knowledge including the similarity measure and the indexing method, 3) the adaptation knowledge and 4) the case base itself.

However, most CBR systems, designed to operate over long periods and that must handle large volumes of data and cases encounter problems in the retrieval and adaptation phases. The latter can be costly in terms of time (Cao et al., 2001). To maintain the quality of system (i.e.) the speed of the retrieval process, maintenance of the case based reasoning system and particularly of the knowledge containers becomes necessary.

1.1. Maintenance of the CBR system

There are two types of maintenance studies, the maintenance policies and the integration of maintenance with case-based reasoning processes (Roth-Berghfer and Iglezzakis 2001). Reinartz et al. (2001) define two phase's steps and tasks (Reviews and Restore) necessary to integrate maintenance into a CBR process. Heister and Wilke (1998) propose an architecture integrating maintenance component in a CBR system that is composed of knowledge container and modeling tools.

During maintenance, the contents of each of the four knowledge containers may be revised (Lu et al., 2009) in order to improve the performance objectives, e.g., the quality of the proposed solution (Arshadi and Jurisica, 2004).

In past decades, a lot of studies have been done in retrieval knowledge maintenance promoting the performance of similarity measurement. Craw et al. (2001) developed a method to optimize CBR retrieval. Bonzano et al. (1997) combines introspective learning for feature weighting in CBR. Feature weights for a set of cases are adjusted dynamically during case retrieval by Zhang and Yang, (1999). Zhang and Yang (2001) maintain the important measures of different features of the case base, by integrating a learning network method for feature weighting within the CBR system. To strengthen the retrieval performance, many works are interested in the case base container and combine problems of feature selection and case organization (Yang and Wu, 2000; Aha and Bankert, 1994; Zhu et al. 2015).

(Salamó and López-Sánchez, 2011a; 2001b) and (Salamó and Golobardes, 2003) focused on feature weighting and instance selection methods based on Rough Set Theory (RST).Those works mainly pay attention to the feature weighting and similarity measurement techniques partially or simultaneously

Regarding the maintenance of adaptation knowledge and the case-base, Shiu et al. (2001) transform the case-base to a set of small case bases each associated with adaptation rules generated by fuzzy decision tree (Lu et al., 2009).

Iglezakis and Roth-Berghofer (2000) suggest that a CBR system cannot be maintained without scanning the case-base. Maintenance activities are activated only via the case-base which plays a major role in the maintenance of CBR systems. This explains why the bulk of the work done in this field is primarily based on Case-Base Maintenance (CBM). In fact, the case-base is the knowledge container that is the most sensitive to changes in the CBR system and its consultation is essential in order to set maintenance operations in motion (Leake and Wilson, 1998).

1.2. Case-based maintenance

CBR systems are large scale case bases. It is thus necessary to keep a compact and competent case base (Salamó and López-Sánchez, 2011a), to maintain the quality of the case base and the speed of the retrieval process (Smiti and Elouedi, 2014). New content cases must be added, existing cases may need to be revised, and out-of-date cases must be deleted; this is a classic example of the case-base maintenance problem (Ferrandiz and Boullé, 2010).

The k-Nearest Neighbors classifier frequently used in Case-based reasoning remains as (i) one of the most well-known algorithms for supervised non-parametric classification (Duda et al., 2001) in Pattern Recognition, data mining and Case based maintenance (Calvo-Zaragoza et al., 2014; Ferrandiz and Boullé, 2010) (ii) as a benchmark for experimental studies in machine learning (Leyva et al 2015). To improve the quality of the case-base, many researchers develop reduction of cases method in the CBR community (Yang and Zhu, 2001, Smiti and Elouedi, 2014, Iglezakis and Roth-Berghofer, 2000), or instance reduction in machine learning community and particularly in instance based learning (Blum and Langley, 1997). Reinartz (2002) proposes a unifying view on Instance Selection, we base on this proposition in order to first draw the link between the work done in instance reduction and in CBM, review it, and then propose a new approach to reconstructing the case-base.

According to Smiti and Elouedi (2014) there are 2 types of CBM policies: **partitioning of the case base** which builds a case-base structure (Yang and Wu, 2000; Cao et al., 2001), and **CBM optimization** which uses an algorithm to delete or update the whole CB (Haouchine et al., 2008; Smyth and McKenna, 1995; Smyth, 1998).

In this study we present, an instance based learning case in the context of classification, where all descriptors are indicated and the solution represents a class.

Different investigations in CBM are reviewed taking into account the framework of Reinartz (2002). Indeed, CBM policies following Reinartz‘s framework can be presented as clustering (also known as case-base partitioning) and prototyping (also known as case-base optimization) steps. We propose a classification of the case-base optimization taking into account selection criteria and search procedure to complete the taxonomies of CBM algorithms presented in machine learning. Indeed, the most fundamental criteria that allow evaluating the case base quality in case-base optimization are the performance and competence.

We are especially interested in the latter in order to achieve the objectives related to search-time problems as well as to reduce case-base size while preserving its quality. Case-base quality depends on a number of criteria that are discussed in the same section. Section 2 ends with a summary of the state-of-the-art study. Several observations and conclusions are given allowing us to propose our CBM method based on the principle that to maintain a case-base, it is necessary to evaluate and optimize its quality by structuring it and auto-incrementing it with new cases while maintaining its structure.

Section 3 describes a novel method based on the structuring of a case base and its auto-increment. Indeed, after having structured the case base, it will be incremented by new cases. However, this integration must be made under specific conditions in order to ensure system quality. Consequently, several questions arise. Which case is to be retained among those solved? How is the case to be indexed? How can it be introduced with respect for the structure of the expert base, to allow a delicate updating of expertise? What is its contribution to the improvement of expert-base quality? An auto-increment algorithm will therefore be proposed in the same section. Section 4 addresses a comparative study of existing methods conducted on particularly significant benchmarks according to competence

and performance criteria. The suggested method is applied in section 5 via a supervised industrial system of pallet transfer (SISTRE).

2. RELATED WORK ON CASE-BASE MAINTENANCE

2.1. Case-base maintenance and instance reduction methods

Maintenance in CBR involves different operations: outdated, redundant or inconsistent cases may be deleted; groups of cases may be merged to eliminate redundancy and improve reasoning power; cases may be re-described to repair incoherencies (Smyth, 1998). Furthermore, case-base maintenance implements policies for revising the organization or contents (representation, domain content, accounting information, or implementation) of the case base in order to facilitate future reasoning (Leake and Wilson, 2000a). Note that this definition considers the information defining an indexing scheme to be an intrinsic organizational component of the case base itself. Case-base maintenance involves revision of indexing information, links between cases, and/or other organizational structures and their implementation (Smyth, 1998; Mykola Galushka and Patterson, 2006).

In this context, maintenance is based on applying update policies for case-base representation and is implicated in their reorganization in order to facilitate future reasoning in response to sets of performance objectives. The state-of-the-art of case-base maintenance relies on the various methods used. Case-base maintenance aims to reduce case search time while improving the performance of the system. Time is reduced by minimizing case-base size or by the partitioning it into several smaller parts.

2.2. Unified framework on instance reduction technique and CBM

Hereafter, we present the different points of comparison between these methods and develop the crucial points. There are several different ways to categorize existing case-base maintenance methods. In this study we take an interest in case size reduction, hence we can draw an analogy with instance-reduction techniques.

2.2.1. Instance reduction technique.

Dai and Hsu (2011) split up instance reduction algorithms into two types: clustering based learning algorithms and instance based learning algorithms. However, Reinartz (2002) has gathered the methods specific to the tasks of instance selection in three steps: Sampling, Clustering and Prototyping. These steps are viewed by Liu and Motoda (2002) as three basic generic components of instance reduction and can be instantiated to individual algorithms.

Sampling is a procedure in which a sample Si is drawn through a random process by which each Si receives its appropriate probability πi of being selected (Liu and Motoda, 2002). A general advantage of sampling is the efficiency of most of its techniques in terms of execution time. However, the existing techniques are not limited only to this method as they are usually combined with others.

Clustering is one approach to finding regularities from unlabeled data (Liu and Motoda, 2002). It is the classification of objects into different groups or, more precisely, the partitioning of a data set into subsets (clusters) so that the data in each subset (ideally) share some common trait, often proximity, according to some defined distance measure. Our method takes this aspect into account.

Prototyping is the process of quickly putting together a working model (a prototype) in order to test various aspects of a design, illustrate ideas or features and gather early user feedback. Prototyping is

often treated as an integral part of the system design process, where it is believed to reduce project risk and cost. Indeed, the prototypes are more condensed descriptions of sets of characteristics.

Prototyping supposes that a simple characteristic can represent the information of an entire subset of characteristics (Pękalska, 2006). The basic idea is, once clusters have been obtained within a larger space, it is then necessary to find among them the most representative prototype of the set. Prototyping thus allows selecting of the relevant cases or the subset of cases which represents the whole set according to a given criterion. This allows removing some cases which results in reducing the case base to a smaller size having the same characteristics as the original.

2.2.2. Case-based maintenance and reduction technique

The case-base maintenance approach can be divided into two policies, one ***concerning case-base partitioning*** and the other one concerning ***optimization***. Policies found in clustering and prototyping steps integrated each one sampling steps. The partitioning policy consists of dividing the case base into several search spaces. This enables selection in an increasing manner those attributes which are rich in information and which can cover the structure of the case base (Yang and Wu, 2000). In contrast, the optimization policy consists of deleting less relevant cases.

2.2.3. Case base partitioning or clustering

In the partitioning of case base, Clustering and feature selection techniques have been successfully applied to CB maintenance (Aha and Bankert, 1994; Yang and Wu, 2000; Perner, 2006).

Zhu et al. (2015) propose a hybrid CBR system composed of neighborhood rough set method as in feature selection technique and by cluster analysis approach, a growing hierarchical self-organizing map (GHSOM), to organize a large case base in a hybrid CBR system.

Smiti and Elouedi (2014) combine DBSCAN (Density Based Spatial Clustering of Application with Noise) and Gaussian-Means algorithms to cluster the case base into small CB's, easier to maintain each one individually. To reduce the size of each partition and preserve maximum competence, outliers and internal cases detection methods are applied. These proposed maintenance policies are named Weighting, Clustering, Outliers and Internal cases Detection based on clustering technique.

Salamó and López-Sánchez (2011a) develop three strategies for feature selection based on rough sets for dimensionality reduction in Case-Based Reasoning classifiers, and propose (Salamó and López-Sánchez, 2011b) an adaptive case-based reasoning model that develops the case base during the reasoning cycle by adding and removing cases. Salamó and Golobardes (2004) introduce a dynamic case base maintenance (DCBM) model that updates the case base based on Reinforcement learning (solving process). This approach integrates the solving algorithm in a case base maintenance process.

To improve performance of CBR systems, Zhu et al. (2015) presents an integrated reduction technique and cluster analysis approach to manipulate the problems of feature selection and case organization in large CBR system.

Hamidzadeh et al. (2014) propose an instance reduction method based on hyper-rectangle clustering. A hyper-rectangle is defined by its min and max points. The clustering algorithm selects subset of instances near or within the boundaries of classes. The size of training set is reduced, improving generalization accuracy.

Ferrandiz and Boullé (2010) propose a new instance selection method, introducing new classification scheme for instance selection named the Voronoi-Based Relabeling scheme,(VBR scheme) for short,

which is a relabeling method relying on Voronoi partitions. The new optimization algorithm finds the best set of instance, by exploiting additive criterion. The kept instances are retained as training data for different classifiers and not only for the kNN algorithm.

2.2.4. Optimization policies or prototyping selection.

Searching for efficient approaches in instance reduction is still an active field of research (Hamidzadeh et al, 2014, Garcia et al, 2012, Lupiani et al 2014, Leyva et al, 2015). A most common technique to reduce the size of the training set, to decrease computational cost and sensitiveness to noise is the use of Prototype Selection method PS (Calvo-Zaragoza et al., 2014). The method proposed by the authors combines the classification accuracy of retaining all the training set and Prototyping Selection methods provided in kNN classification. This PS method first reduces the training set by using an algorithm and performs the classification of the new element. Prototype Selection PS ~~algorithms~~ allow a faster Nearest Neighbor classification by keeping only the most profitable prototypes of the training set.

Prototypes can be defined as generalized numerical examples (Salzberg, 1991) or tuples composed by both numerical and categorical values like hyperrectangle.

Calvo-Zaragoza et al. (2014) list representative sets of PS algorithms, all based on Nearest Neighbor NN published in the literature, such as Condensing Nearest Neighbor (CNN), Editing Nearest Neighbor (ED) etc. Those algorithms dedicated to the optimization policies of CBM, are classified by different authors (Fazzolari et al., 2013; Garcia et al., 2012, Verbiest et al., 2013).

2.2.5. Taxonomy of optimization policies or prototyping selection methods.

2.2.6. Prototyping selection method

Some authors have proposed different taxonomies of CBM algorithms (Lupiani et al., 2014; Garcia et al., 2012). Garcia et al. (2012) describe the different properties issued from variable selection and adapted to the PS methods, the properties specific to the prototyping selection methods and at the end the properties that can influence the results of an instance selection algorithm in combination with a given classifier.

Fazzolari et al. (2013) according to Garcia et al. (2012) classified the Prototype Selection PS method and the Training Set Selection TSS methods using common proprieties. We retain the most relevant properties for the methods of PS: evaluation of search, direction of search propriety, type of selection.

- **The Evaluation of search**: according to the strategy used to add or remove instances in the subset S. Garcia et al. (2012) define (i) a filter strategy "when the KNN rule is used for partial data to determine the criteria of adding or removing and no leave –one-out validation scheme is used to obtain a good estimation of generalization accuracy". (ii) And the wrapper strategy "when the kNN rule is used for the complete training set with the leave-one-out validation scheme.
- **The Direction of search** propriety is defined as well in the framework of feature selection algorithm in data mining community by (Liu and Motoda, 2002; Shiu et al., 2001; Blum and Langley, 1997) and refined by adding Three items (Batch search, mixed search and fixed search a subfamily of mixed search (i.e. item that we don't retain).
- **Search Direction** (Starting point of the best subset case search): there are a variety of directions in which search can proceed.

- Forward Selection (FS) or incremental search consists of building a reduced case base by successive addition, starting from an initially empty case base, according to a criterion to be maximized. There are two categories of methods: one maximizing the competence criterion and the other the performance criterion
- Backward Elimination (BE) or decremental search: the case-base will be screened entirely when its size reaches a certain threshold, usually followed by the process of case deletion
- Batch search involves deciding if each instance meets the removal criteria before removing any of them.
- Mixed search begins with preselected subset and can iteratively add or remove any instance which meets the specific criterion.

- **The type** of search carried out by the PS algorithms that seek to retain set of point with respect to the decision boundaries (border instances, central instances) can be of 3 types: condensation, edition and hybrid methods.
  - ***Condensation methods*** try to obtain a consistent subset by removing unnecessary instances that will not affect the classification accuracy on the training set.
  - ***Edition methods*** aim to remove noisy instances, allowing the classifier to increase its accuracy.
  - ***Hybrid methods*** search for a subset in which both noisy and unnecessary instances are concurrently eliminated.

The taxonomy of Garcia et al. (2012) takes into account the type of selection, the search direction and the evaluation search. The figure 1 illustrates this classification.

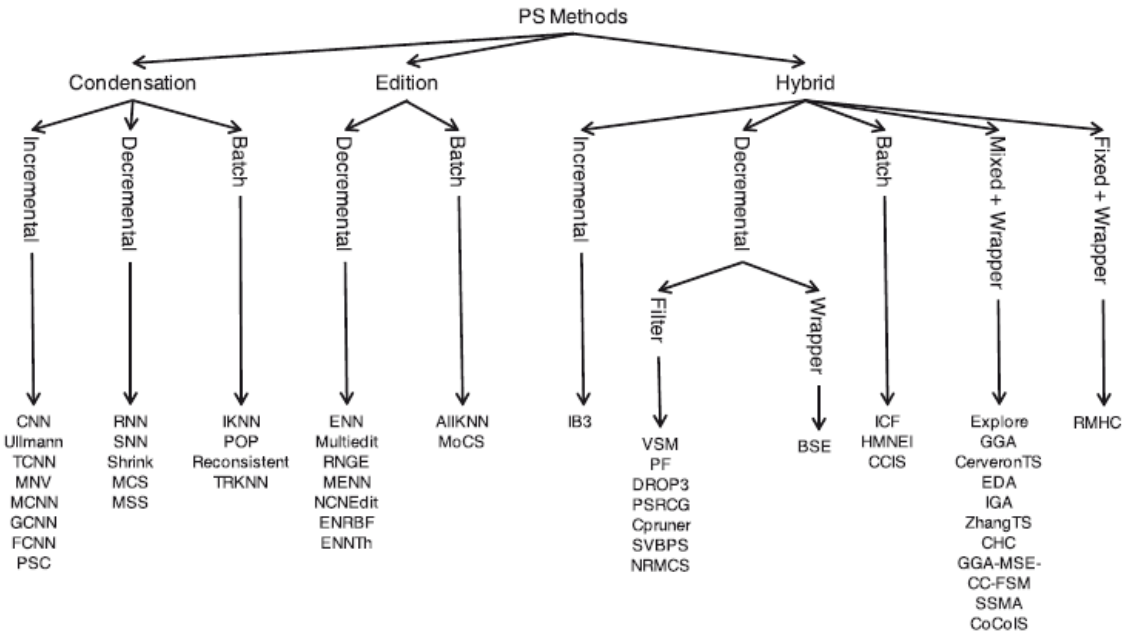

Fig.1. Prototype Selection (Garcia et al. 2012)

Leyva et al. (2015) propose a classification of the PS algorithm taking into account the type of selection and defines (i) the condensation method like the methods of the family of Condensed Nearest Neighbor (CNN) Selective Nearest Neighbor (SNN) Minimal Consistent Set (MCS) and Fast Nearest Neighbor Condensation (FCNN). These methods are very noise sensitive. (ii) The edition method follows the family of Edited Nearest Neighbor (ENN); Repeated ENN (RENN).Theses methods reject the instances that affect in classification with their neighbor- hoods like. They are characterized by being good noise filters, and achieve little reductions in the number of instances (Leyva et al., 2015)

(iii) The hybrid methods are a combination of both edition and condensation strategies like Instance Based Learning3 (IB3), decremental Reduction Optimisation Procedure DROP3, Iterative case filtering ICF etc... The IB3's algorithm use previous selected instances to build incrementally selection to classify. It retains instances misclassified, and selection the instances having a best accuracy of the member of the class and eliminate the instance with the worst accuracy.

Verbiest et al. (2013) classified the IS method based on the type of selection and the evaluation of search. The authors propose a FRPS Fuzzy Rough Prototype selection which is a wrapper-oriented edition method. The instances are ordered according to a measure based on fuzzy rough set theory. A wrapper approach is used to prune instances and a kNN algorithm to classify the instances.

2.2.7. Optimization policies

The main principal of ~~the review of~~ the case-based optimization policies is to build a maintenance model that represents the implicit information of the case. We propose to classify the CBM algorithms based on two additional proprieties that are defined in the feature selection framework: the search procedure and the selection criteria:

- Search procedure (subset evaluation function): there are three search procedures: complete, heuristic and random generations. Each search procedure corresponds to the nature of the case base and to the chosen method for case selection.
- Selection criteria: represent the evaluation criteria of the case base. The various existing methods can simultaneously use one or more criteria.
- Stop criteria: generally, case-base maintenance methods have a stopping criterion of selection or removal of case-base cases. This criterion plays a threshold role and it is important to determine it well.

First we describe these criteria adapted to the case-base.

2.3. The Quality criteria for case-base evaluation

An effective case base can answer as many queries as possible, efficiently and correctly. Several criteria allowing the evaluation of case-base quality have been proposed in the literature. We can quote inconsistency, redundancy and abstractness, introduced by Racine and Yang (1996), and relevancy, introduced in (Blum and Langley, 1997).

However, the important criteria contributing to the evaluation of a case base are: competence and performance. These criteria can be based on coverage and reachability notions.

Performance: Case-base performance is measured by the answer time necessary to compute a solution for case targets (Smyth and Keane, 1995). This measure is bound directly to adaptation and result costs.

Competence: Case-base competence is measured by the range of problems that can be satisfactorily solved (Smyth and Keane, 1995). Case-base competence thus represents the case coverage which it contains. Several strategies which make it possible to measure the case coverage have been proposed in the literature (Smyth and McKenna, 1999, 2001a, 2001b).

Indeed, competence is based on the two notions of "coverage" and "reachability".

2.3.1. Coverage and reachability notions

Given a case base **CB** be $\{c_1,.., c_n\}$ and a set of target cases "T" $\{t_1,.., t_r\}$. A case "c" is composed of two parts: problem and solution.

Case c = { Ps, Ss } Target t = { Pt, ? }

Coverage of a case c: is the set of target problems that it can be used solved by its retrieval and adaptation.
Reachability of a case: is the set of cases that can be used to solve the target problem.

Coverage (c∈CB) = {t ∈CB: Solves(c, t)} (1)
Reachability (c∈CB) = {t ∈CB: Solves(t, c)}

Thus, coverage and reachability is formalized by Smyth & Keane (1995) for c∈ CB

Coverage (c) = {Adaptable (c, t)} (1bis)
Reachable (c) = {Adaptable (t, c)}

In the case of instance based learning where there is not the adaptation phase, this definition become

Coverage (c) = {Similar threshold (c, t)} (1ters)
Reachable (c) = {Similar threshold (t, c)}

Good competence of the case base CB means that its coverage is high and that its reachability is low.

Moreover, we note that the definition of coverage and reachability, introduced by Smyth and Keane (1995), takes into account both the problem and solution parts of case.
Smyth ad Keane (1995) assume that the case-base itself is a sample of the underlying distribution of target problems.

2.3.2. On line problem solving

On line, the CBR objective is to solve a target case and find the solution St to the problem Pt. We do not have the solution target = { Pt, ?} unlike at the offline step (structuration of the case-base) where the set of target case is a distribution of case-base
Unlike the offline step (the case-base structuring) where the set of target cases is a distribution of the case-base, the target solution at this step is not known.
During the problem solving step, a potential solution is assigned to each target. It is named target*={Pt, St} . Once the target* is analyzed, it is integrated in the case-base where it either becomes a case or it is deleted.

2.4. Reviews of case-based optimization policies

As Dai and Hsu (2011), we address the strategies of sample selection developed within the case-based optimization policy, namely: forward selection (incremental search) and backward elimination (decremental search).

2.4.1. Forward Selection (FS).

In the first category of FS, Smyth and McKenna present a method that uses an explicit case competence model based on notions of coverage and reachability. Their relative coverage (RC) metric provides a precise measurement of **competence contributions** for individual cases.

$$RC(c) = \sum_{c' \in CoverageSet(c)} \frac{1}{ReachabilitySet(c')} \qquad (2)$$

With definition of coverage and reachability (1)

CoverageSet (c∈CB) = {c' ∈CB: Solves (c, c')}
ReachabilitySet (c∈CB) = {c' ∈CB: Solves (c', c)}

The RC metric, associated with the condensed nearest-neighbor (CNN) algorithm, allows successive retention of only those cases which have not yet been solved by a case that has already been retained, in order to obtain a new reduced case base (Smyth and McKenna, 1999). Consequently, the selected cases make an important contribution concerning case-base recovery.

Yang and Zhu (2001) describe a case-addition algorithm for case-base compaction that uses a problem-neighborhood model of case coverage. The cases based on benefit/usefulness are successively added to the case set so far retained.

The interesting part of these methods is the use of models and metrics that make it possible to guide the case-base size reduction by preserving good **competence**. The cases are ranked by the metrics in order to add the most interesting cases to the reduced case base. To obtain a reduced case base, computational time becomes greater. Indeed, for each added case it is necessary to re-examine the entire original case base.

In the second category of methods maximizing the **performance criterion**, Leake and Wilson developed a Relative Performance (RP) metric aimed at assessing the contribution of a case to the adaptation performance of the system (Leake and Wilson, 2000b). The RP value for a case rejects how its contribution to adaptation performance compares to other cases. Likewise, another metric was developed concerning a Performance Benefit (PB) metric estimating the actual numerical savings that the addition of each case provides. Yet, in comparison to the RP-CNN and PB-CNN methods, RC-CNN produces a better rate of case-base size reduction, whereas the former methods give a better result in adaptation cost.

The FS strategy includes only one criterion. However, the majority of these strategies use measures of performance or competence in combination with the CNN algorithm. Indeed, these strategies classify the cases in the initial case-base according to the used measure value used (RC, RP or PB) and they then begin the process of a case-addition within a case base that is initially **empty** according to the CNN algorithm. Both RP-CNN and PB-CNN strategies are very close in terms of results concerning the performance of the case-base. These last two strategies only concern performance whereas the RC-CNN strategy and the Yang and Zhu's algorithm (Yang and Zhu, 2001) are only concerned with case-base competence. In fact, they aim to reduce case-base size while maintaining its competence.

In summary, the FS strategies treat case-base quality by exploiting only one criterion that is optimized according to reduction of case-base size. The majority of these strategies (RC, RP and PB) combine the CNN algorithm with either a competence or performance measure.

The FS strategy results depend on the first case selected, which is randomly selected in CNN, or is case-dependent in RC, RP or PB measure. If the first case is badly chosen (due to a skew of measure), the reduced case base is affected and is of low quality. Consequently, in this study we focus on the backwards elimination (BE) methods, which, though costly in computing time, provide an overview of the case base.

2.4.2. Backward Elimination (BE).

From a given case base, this strategy emphasizes cases according to its criteria so as to reduce their number and thus shrink the case base to a specific number of cases. Evaluation criteria such as competence, redundancy and inconsistency, have been used in different methods, which will be explained below.

In the BE method, the case-base will be screened entirely when its size reaches a certain threshold, usually followed by the process of case deletion. There are two kinds of BE methods: 1) suppression methods for using case-base screening and, 2) methods from the case categorization.

Suppression methods

The first BE method, Random Deletion (RD) is a very simple baseline method, when a case is randomly selected and deleted once the case-base size exceeds a predefined limit (Markovitch and Scott, 1988). Another method, **Ironically (Ir),** is slightly more complex. It calculates the frequency with which each case is retrieved and deletes those which are not frequently accessed in the case base (Minton, 1990).

BE methods have thus evolved and take into account a quality criterion in their study. Among these methods we find Utility Deletion (UD) which is based on Minton's utility metric and chooses a case item for deletion based on an estimate of its performance benefits (Smyth and Cunningham, 1996).

A utility metric is defined by Minton as:

$$metric = (ApplicFreq \times AverageSavings) - MatchCost \qquad (3)$$

which takes into account the cost of maintaining case and the average savings multiplied by application frequency.

The **utility** problem manifests itself as a trade-off between the solution quality associated with large case base and the efficiency problem of working with a large case base. Furthermore, solution quality increases with case-base size (Smyth and McKenna, 1998).

Lastly, there exist other methods such as the Deletion Based on Redundancy and Inconsistency method which are endowed with two modules of detection, one with redundancy and the other with inconsistency. After a series of tests using these two modules and concerning each case found in the base, the specific cases may, after user approval, be either removed or kept (Racine and Yang, 1996). Other methods take into account other notions, such as density in Deletion Based on Case-Base Size and Density.

The first of these, proposed by Smyth and Keane, studies case-base size and the density distribution of the cases contained in the base. It attempts to respect the homogeneity of case-density size (Smyth and McKenna, 1998).

RNN (Gates, 1972) the reduced nearest neighbor algorithm keeps removing instances until no more misclassification is generated by the remaining instances. It is hence better than CNN in terms of instance reduction percentage and classification speed. An extension of this algorithm is the Generalized Condensed Nearest Neighbors (GCNN), it is the same as CNN, but assigns instances which satisfy an absorption criterion (calculated in terms of the nearest neighbors and enemies (the nearest instances of other classes)). An instance that is absorbed using the absorption criterion could better reduce the size compared to CNN algorithm.

Wilson and Martinez (1997) introduced a notion of association where an instance is an associate of another instance if it is a one of its k nearest neighbors. This notion is applied in a family of five decremental reduction optimization procedure algorithms called DROP1-DROP5. Drop3 uses ENN to remove the noisy instance before executing drop1 and presents a best performance (Wilson and Martinez, 2000 ; Hamidzadeh et al., 2014).

Dai and Hsu (2011) propose three Reverse Nearest Neighbor Reduction RNNR algorithm versions based on Reverse Nearest Neighbor (RNN) and selecting the better RNR-L1 that achieve higher accuracy than ICF and DROP3.

2.4.3. Batch search.

The ICF method, whose type selection is hybrid was introduced by Brighton and Mellish (2002a), uses an algorithm which iteratively removes a case whose absence produces better results than if it were retained. This algorithm also uses coverage and reachability as selection criteria. It repeatedly uses a deletion rule that removes cases whose reachability size is greater than that of the coverage until the conditions of the rule are no longer satisfied.

The majority of these methods do not give satisfactory results concerning the optimization of case-base size according to the studied criterion. Moreover, there are some methods which are difficult to implement, and those which are easy to implement do not produce convincing results. We are interested particularly in the ICF method because it produces better results than the others (Brighton and Mellish, 2002a).

ICF outperforms alternative local methods such as ENN, CNN, IB3 and DROP3. It is thus the best candidate for representing **local methods** (Ferrandiz and Boullé, 2010).

Leyva et al. (2015) argue that the ICF method presents some drawbacks but "it is an interesting method that opened the promising field of using LSs in Instance Selection, whose study may be a source of inspiration for new proposals in this field." The authors propose three instance selection IS methods based on local sets, which follow different and complementary strategies. Indeed the authors resolve the problem of instance reduction (maximizing accuracy and reduction size of case base), as a bi-objective problem and use a local set concept to develop three IS methods. These methods differ in the selection strategy and presents complementary results (the first maximizes the accuracy, the second one targets the reduction of size and the third makes a compromise between these objectives.

2.4.4. Backward Elimination categorization

2.4.4.1. Case base competence modeling

The second kind of BE is a categorization method used for modeling the case-base competence as proposed by Smyth and Keane (1998; 1999; 2001a; 2001b). The cases included in a case-base are categorized according to their competence. The key concepts in categorizing cases are the coverage and reachability previously discussed.

The methods developed in this area generate a set of target cases so that case base is categorized. Two basic hypotheses underlie these models: in the first the case base corresponds to a sample target or to potential cases, and in the second the space problem is regular, which means that similar problems have similar solutions.

Footprint deletion is the strategy used to remove irrelevant cases, thereby guiding the case base towards an optimal configuration (in the sense that competence is maximized while size is minimized).

The case categories described above provide a means of ordering cases for deletion in terms of their competence contributions. Auxiliary cases are selected for deletion before support cases, which are chosen before spanning and pivotal cases. The optimal case-base can be constructed from all the pivotal cases plus one case from each support group. This strategy is not designed to eliminate the need for performance-based methods such as utility deletion (Smyth and Keane, 1995).

Footprint Utility deletion is the hybrid strategy between footprint deletion and utility deletion. First, the footprint method is used to select candidates for deletion. If there is only one such candidate then it is deleted. However, if there are numerous candidates, rather than selecting the one with the least coverage or largest reachability set, the candidate with the lowest utility is chosen (Smyth and Keane, 1995).

2.4.4.2. Specific case categorization

These two previous methods use a specific case categorization. Four categories of cases are considered:

Pivotal Cases: a case is pivotal if it is reachable by no other case but itself. Its deletion directly reduces the competence of a system.

Pivot(c) iff Reachable(c) – {c} = ∅

Spanning Cases: spanning cases do not directly affect competence. They are so named because their spaces of coverage link (or span) regions of the problem space that are independently covered by other cases.

Spanning(c) iff ¬**Pivot (c)** Λ coverage(c)∩ $U_{t\in Reachable(c-\{c\})}$Coverage(c) ≠ ∅

The spanning is intra-class if t and c have as solution the same class. Otherwise it is inter-class.

Support Cases: support cases are a special class of spanning cases and, again, do not affect competence directly. They exist in groups, each support providing coverage similar to the others in a group. While the deletion of any one case from a support group does not reduce competence, the removal of the group as a whole is analogous to deleting a pivot, and does reduce competence.

Support(c) iff ∃t ∈ Reachable(c) – {c}: Coverage(t) ⊂ Coverage(c)

Auxiliary Cases: a case is auxiliary if the coverage it provides is subsumed by the coverage of one of its reachable cases. Auxiliary cases do not affect competence at all and their deletion only reduces the efficiency of the system.

Support(c) iff ∃t ∈ Reachable(c) – {c}: Coverage(c) ⊄Coverage(t)

The methods of BE strategy make it possible to optimize only one quality criterion of the case-base at the expense of other criteria, the competence for RD and the performance for UD. Although FUD takes competence and performance criteria into account in its method, no evaluation of the case-base has been made concerning its performance. In contrast, the development of redundancy and inconsistency criteria (DRI) provides good competence but bad performance. Consequently, we will

jointly study a criterion for case selection that takes into accounts both performance and competence to ensure a higher quality case base.

It is noteworthy that the working sample relating to target cases is exploited to evaluate case-base quality according to the criterion under study. This sample of cases represents the entire case base for the methods RD, Ir, UD, DRI and DSD, whereas in the other two methods, FD and FUD, the sample is only represented by a subset randomly selected from the case base.

In the first case, the study is of greater interest since the entire case base is scanned and therefore all possibilities are represented. However, computation time is greater than in the second case. We can also specify that all methods make use of a heuristic search procedure except RD which consists of a random procedure as its name implies.

The advantage of these methods is the case categorization used for their deletions. Clear and effective, they also produce good results in execution speed once implemented. Their disadvantage is that their setup is costly in computing time. Moreover, Yang and Zhu (2001) have shown that these categorization methods do not guarantee the preservation of competence. Cases having a strong coverage can be removed.

The contribution of this paper is in the area of backward elimination strategy. Indeed, we propose a methodology for case-base optimization by deleting the lowest quality cases in the case base. The case quality is being assessed on the basis of competence and performance criteria.

3. THE PROPOSED METHOD

3.1. General cycle of the proposed case base maintenance method

The proposed case-base maintenance method is comprised of two stages (see Fig. 2):

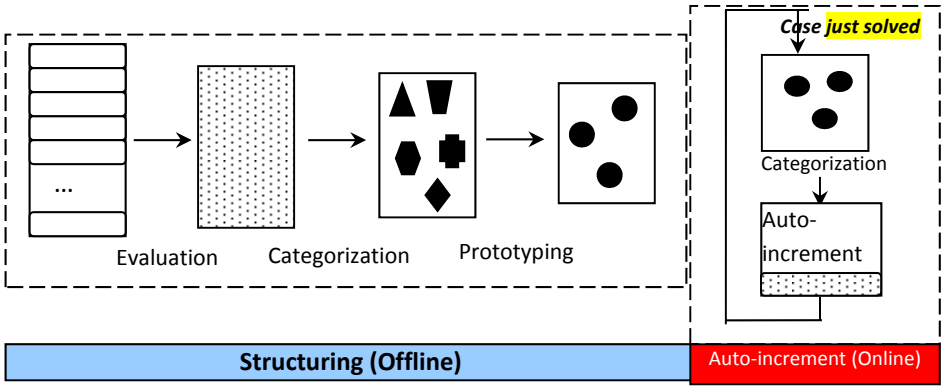

Fig.2. Proposed case-base maintenance cycle

Stage 1: "structuring the case-base"

This stage is composed of three steps: (i) evaluation of the case base, (ii) categorization of different cases (clustering) and (iii) prototyping.

The step of assessing case-base quality is undertaken by the joint use of the two criteria of competence and performance. Competence is characterized by the calculation of case-base coverage and reachability. Performance is characterized by the accuracy of the case base and the number of cases. This step is for the preparation of case categorization, an evolution of the competence model of Smyth

and McKenna (1998), with five case categories defined: pivotal, support, auxiliary, inter-class spanning and intra-class spanning. The next step, "prototyping", is for the selection of the most representative cases, which, in turn, depends on case categories.

Stage 2: auto-increment of the case-base"

This stage makes it possible to incrementally add to the case base new cases that have just been solved by the CBR cycle.

From the average recovery cases of the restructured case base, the coverage and reachability of the new case is calculated, taking into account the problem and solution parts of the case in comparison to the case base. This requires defining of measures that take into consideration the coverage and reachability of the two case parts of problem and solution. We will rely on these measures in order to decide to increment or not the new cases in the case base. It would thus be possible to respect the structure of the previously set up case base. If the solution is reachable, we focus on the problem part of the new case in order to account for the structure of cases previously defined.

The adopted approach relies on the following points.

Approach: clustering + prototyping (prototype selection)

The proposed approach consists of gathering into different categories cases considered to be similar according to their coverage and reachability. A prototype is then selected for some categories by choosing the case which represents its group as well as possible.

Search Direction: Backward Elimination

From a complete case base, cases are removed to reduce the case base according to a set of criteria. This suppression is carried out using a categorization algorithm as well as a competence measure.

Search Procedure: heuristic + random

For each case in the case base, we estimate its coverage and reachability from a subset of cases randomly drawn, that will represent the problem space, i.e. the cases which will be potentially reachable by the source cases.

Selection Criteria: competence and performance

The objective of the proposed method is to preserve the competence of the case-base while optimizing its size and maintaining good performance. Competence is quantified by a measure (CM) that keeps with the two basic concepts: coverage and reachability.

To achieve high competence in the case base, it is necessary to maximize the coverage of cases and minimize their reachability. As for performance, it depends on the accuracy of the case-base cases (good classification) according to their number (storage).

$$CM(c) = \frac{Vc(c)}{Vr(c)} \quad \textbf{(4)}$$

V c(c) = Cardinality of the c case covering set.
V r(c) = Cardinality of the c case reachability set.

Stopping Criterion: according to a preset global competence measure

A value is determined from the CM measure making it possible to determine when to stop deletion. This measure reflects the global competence value of the complete set of the reduced case base which is equal to the sum of the CMs of each case, divided by the number of cases in the case base. Thus, when the global competence value is equal to "1" and the coverage of each case in the case base is alone in its own class, then deletion stops.

$$GlobalCompetence(CB) = \frac{\sum_{i=1}^{n} CM(c_i)}{n} \qquad (5)$$

Where n: is the number of cases in the case base.

The proposed method is composed of two steps consisting, on the one hand, of a way to structure the case base, and, on the other hand, of the means to dynamically ensure its auto-increment.

3.2. Structuring the case-base

As mentioned above, structuring is achieved by backward elimination and is comprised of: evaluation, categorization and prototyping.

3.2.1. Evaluation.

The evaluation step is based on the case categorization developed in Smyth and Keane (1995) and adapted for this study to select representative cases. It can work by means of an algorithm associated with the Competence Measure (CM), a methodology that deals with two axes. The cases are treated first: Smyth's categorization is used to guide deletion and the CM metric to construct a compact competent case base. The labeled instances are treated second: using the Smyth categorization, the spanning cases are divided into two subcategories, with the CM metric guiding the deletion.

To calculate the coverage and reachability of the cases, the k-nearest neighbor (knn) algorithm is then applied by using the Euclidian Distance in the numerical instance and the Hamming distance in categorical variables.

A case is composed of a Problem part and Solution part.

$$Case = (P_s, S_s)$$

$$P_s = (d_1, d_2, \dots, d_i, \dots . d_{m-1});\ S_s = (d_m)$$

The similarity in the case of numerical instances is equal to

$$Sim\left(d_i^k, d_j^k\right) = \frac{1}{1 + \sqrt{(d_i^k - d_j^k)^2}}$$

However for the categorical case we have.

$$Sim\left(d_i^k, d_j^k\right) = \begin{cases} 1 & if\ d_i^k = d_j^k \\ 0 & if\ d_i^k \neq d_j^k \end{cases}$$

And when there is a mixture of instances, we discretize the numerical variables in order to handle only the categorical ones.

$$Sim(case_i, case_j) = \sum_{k=1}^{m} Sim\left(d_i^k, d_j^k\right)$$

The algorithm below (Algorithm 1) illustrates the calculation of coverage values (Vc) of each case-base case

---

1. **Initialization:**
   CoverageListCase$_i$=∅
   Vc$_i$=0, i=1..n
2. **For** i=1..n (each case$_i$) do // n: number of cases in CB
3. **For** j=1..n (each case$_j$) **do**
   4. **If** sim (case$_i$, case$_j$) > Threshold then // case$_i$ covers case$_j$
   5. Vci = Vci + 1 // the value of coverage is incremented
   6. Select the index (case$_j$) and Vc$_{ij}$
   7. Update the List-associated_casei
   8. CoverageListCase$_i$: (Vci, index (case$_j$)j∈{1..n})

   EndIf

   EndFor

EndFor

Algorithm 1. Calculation of Vc

Two examples of CoverageListCase$_j$:
CoverageListCase$_1$: (Vc$_1$ = 3, case2, case12, case20)
CoverageListCase$_5$: (Vc$_5$ = 2, case7, case16)

The algorithm of V_c calculation results in forming two groups of cases. The first group is related to the case base containing "case$_i$" cases (i=1..n), the second to the entire "case$_j$" set of target cases (j=1..n) which is equal to the number of cases in the case base (hypothesis: a source case is a potential target case). For each "case$_i$" of the case base, the algorithm scans the entire target case "case$_j$". If the similarity between a case$_i$ and a case$_j$ exceeds a preset threshold, then the case$_i$ covers the casej. The algorithm increments a counter dedicated to the case$_i$ coverage value "Vci" and simultaneously memorizes the index of case$_j$. Thus, when the scanning of the target-base for the first case$_i$ has ended, a couple is obtained whose coverage value contains a set of case$_j$ indices that are covered by the case$_i$. The algorithm repeats this operation "n" times corresponding to the number of cases in the case base.

The algorithm below (Algorithm 2) illustrates a reachable list associated with case$_j$.

---

1. **Initialization:**
   ReachableListCasej=∅
   Vrj=0, j=1..n
2. **For** j=1..n (each case$_j$ ∈ CB) **do** // n: number of cases in CB
3. **For** i=1..n (each case$_i$ ∈ CB) **do**

   // ReachableListCasej = (Vc$_i$, $\sum_{k=0}^{l}$ index(case$_k$), l: number of cases covered by case$_i$)

4. **For** k=1..l (each case$_i$ ∈ CoverageListCasei) **do**
   5. **If** index(case$_i$)= index(case$_k$) **then** // case$_j$ is reachable by case$_i$
   6. Vr$_j$ = Vr$_j$ + 1 // the value of reachability is incremented
   7. Upgrade ReachableListCasej

   EndIf

EndFor

EndFor

Algorithm 2. Calculation of Vr

Two examples of ReachableListCase$_j$:
ReachableListCase$_2$: (Vr1 = 1, case1)
ReachableListCase$_7$: (Vr5 = 3, case5, case13, case21)

This algorithm begins with the fact that there are already coverage values for the different case$_i$ in the case base as well as the index sets of covered cases. As in the Vc algorithm, two groups of cases are exploited: a case group in the case base and a group of target cases containing exactly the same source cases. Then, for each target case, the algorithm compares its index with all of the indexes of the first case$_i$ of the case base. If there is equality, then case$_j$ is reachable by the case$_i$ and so on, for all the cases of the case base. We obtain at the end of the first scan of case$_i$ the reachability value of casej and the reachable indexes of case$_i$. The algorithm repeats this operation «n» time.

3.2.2. Categorization

The categorization step allows the various categories for cases and labeled instances to be defined.

A- Treatment of Cases

Cases are represented by a set of attribute-values. The CM incorporates two properties: coverage and reachability (see equation 2) and makes an individual contribution to the case competence in relation to the size of the latter's coverage set, while attributing to each coverage and reachability case a value that we shall name coverage value "Vc" and reachability value "Vr".

For a case-base to have good competence its coverage ratio must be high and its reachability rate must be low. Consequently, the CM is used to guide the deletion of cases in the case base by favoring cases with a high CM value and deleting those with smaller CM value. Our method therefore consists of reducing case-base size while maintaining maximal competence. The case categories will be determined by CM metric.

The CM value can be calculated using the Vc and Vr values and thus lead to case categorization. The properties that allow this categorization are shown in Table 1.

Table 1. Properties of the case categories.

Type of case	Vc(ci)	Vr(ci)	CM(ci)
Auxiliary case	>1	= Vc(ci)	1
Support case group	>1	>1	Same values
Spanning case	≥1	>1	≤1
Pivotal case	1	1	1

The CM value determines the choice of pivotal cases. It is very important that a pivotal case be preserved because its deletion directly reduces case-base competence. Moreover, a representative having the highest CM value from each support case group is kept. In contrast, auxiliary and spanning cases arriving below a certain competence threshold do not affect competence and can thus be deleted. It is noteworthy that auxiliary cases are the least important as they make no direct contribution to competence. They are followed in ascending order of importance by the support cases, then the spanning cases, and finally the pivotal cases, the most important ones. The following is a case-base

example comprised of four cases showing the coverage and reachability space of the c1, c2, c3 and c4 cases (Fig. 3).

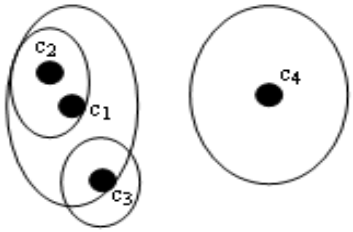

Fig.3. Example for coverage and reachability

Coverage(c1)={c1, c2, c3}	→Vr(c1)=3	Reachable (c1)={c1, c2}	→Va(c1)=2
Coverage(c2)={c1, c2}	→Vr(c2)=2	Reachable (c2)={c1, c2}	→Va(c2)=2
Coverage(c3)={c1, c3}	→Vr(c3)=2	Reachable (c3)={c1, c3}	→Va(c3)=2
Coverage(c4)={c4}	→Vr(c4)=1	Reachable (c4)={c4}	→Va(c4)=1

As a result: CM(c1) = 1.5, CM(c2) = 1, CM(c3) = 0.5, CM(c4) = 1.

Case categorization and deletion are determined by the following rules:

Vc(c2) = Vr(c2) and Vr(c2) > 1, CM(c2) = 1	→ auxiliary case	→ remove case c2
Vc(c3) = 1, Vr(c3) > 1, CM(c3) < 1	→ spanning case	→ remove case c3
Vc(c4) = Vr(c4) = CM(c4) = 1	→ pivotal case	→ retain case c4

In so doing we obtained the deletion of two cases (c2 and c3) and a case base containing cases c1 and c4. By recalculating the CM value of each case, we find that CM(c1) = CM(c4) = 1 with Vr(c1) = V r(c4) = 1 and Vc(c1) = Vc(c4) = 1.
Consequently we found a reduced case base with two cases forming an optimal case base with two pivotal cases.

B- Treatment of Labeled Instances

When an instance-base is treated containing instances with their classes, the CM metric and the Smyth categorization are always needed. However, two subcategories are determined in the spanning cases. The first subcategory concerns inter-class spanning cases and the second, intra-class spanning cases.
An inter-class spanning case of a given class (for example class1) is one which is partially covered by another case belonging to another class (class2). An intra-class spanning case is one which is partially covered by another case pertaining to the same class.
Figure 4 shows an example of a pivotal case (c1 ∈ class2), an inter-class spanning case (c2 ∈ class1), an auxiliary case (c3 ∈ class1), an intra-class spanning case (c5 ∈ class3) and a support group comprised of three support cases (c7; c8; c9 ∈ class4).

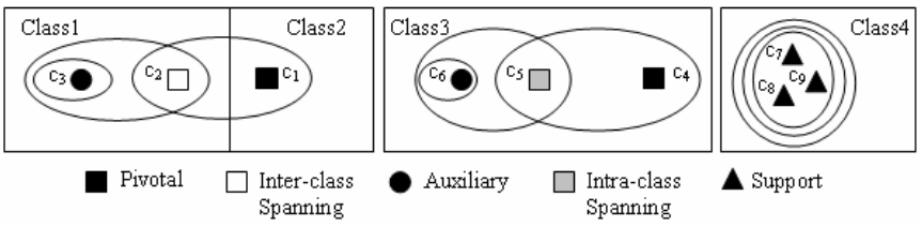

Fig. 4. Case categories

$Case_2$ belongs to class1 but it is covered by the pivotal case1 pertaining to class2. This case should not be removed because it contributes to the competence of class1. In addition, a covering threshold is set at which inter-class spanning cases are deleted. If the covering of case1, which is in relation to $case_2$ (pertaining to the same class), is higher than a preset threshold, then case2 is removed. This is also valid with the first scenario (treatment of cases). This threshold is given to prevent the removal of cases that make a significant contribution to covering and that are reachable by a small part of their covering space. Consequently, the intra-class spanning case definition to be removed is as follows:

Given that (c_1, c_2) ∈ same class

- Intra-class Spanning Case: Spanning (c_2)

$$iff\ \neg Pivot(c_2) \wedge \text{Recovery}(c_2) \cap\ U_{C1\ \in\ \{(\text{threshold})Reachability(c2)-(c2)\}}\ \text{Recovery}(c_2) \neq \emptyset$$

The following algorithm (Algorithm 3) shows the removal process of cases according to their categories. It concerns the instance bases containing the labeled instances.

---

1. **For** each case ∈CB to calculate Vc and Vr
2. Associate at each index (case) its coverage-set and reachability -set
3. Determine the cases categories
4. **If** auxiliary Case **then** Remove all cases

   ElseIf Support Case **then**

   Classify these cases according to their MC values in an increasing way

   For each Support Group **Do**

   Remove all the cases except the one that has greatest MC value

   EndFor

   ElseIf Intra-class spanning cases **then**

   Remove all cases except that which has "Vc < threshold"

   ElseIf Inter-class spanning cases **then**

   Retain

   ElseIf Pivotal case **then**

   Retain

   EndIf

Stop when each case covers only its own case among the existing cases in its class and the CompetenceGlobal(BC)=1.

Algorithm 3. Structuring algorithm For each case to calculate Vc and Vr

The algorithm starts by calculating the two values Vr and Va. It then determines the case categories beginning with the support cases in order to carry out the suppression. It classifies cases according to their Vr value for each support group, keeping only the one which has the greatest Vr value. Then it deletes all auxiliary cases, except the inter-class ones with high CM values. Finally, it keeps all pivotal cases.

Once the cases have been treated, the two subcategories of the spanning cases are no longer taken into consideration.

3.2.3. Prototyping

This step allows for the selection of relevant cases. Selection is performed by deleting the auxiliary cases, deleting intra-class spanning cases according to certain conditions and deleting all support cases except the one with the greatest coverage in its support group, so as to keep inter-class spanning cases as well as pivotal ones. This selection is crucial because the quality of the case base depends on it. Moreover, it allows the number of cases to be reduced, thus rendering the retrieval and adaptation phases effective.

Moreover, in order to maintain the case base throughout the life cycle of the CBR system, we propose an auto-increment method which preserves the structuring previously established with the addition of new cases to the base.

3.3. Auto-incrementing the case base

In order to increment a case in the case base, it must first be evaluated and categorized. It is then trained within the case base by means of a specific algorithm.

3.3.1. Evaluation and categorization of the new case.

After setting up the structuring of the case base through case categorization, the aim of the following study is to make the base evolve in an incremental (dynamic) way while respecting its structure. This evolution requires the introduction of cases into the case base under specific conditions directly related to the quality of the case base.

The two criteria allowing assessment of this quality were discussed in section 2.1, namely competence and performance. In the off line step, when a target case arrives, its solution is not available. We must differentiate to calculate reachability and coverage of the problem and solution part.

Regarding competence, in our study, case coverage will concern both the problem and solution parts. Coverage of the problem part, denoted by (Vcp), will be considered first, followed by coverage of the solution part, denoted by (Vcs). The notations Vrp and Vrs will be assigned respectively to the reachability in each part.

Vrp: is the reachable cardinal of the problem part.
Vrs: is the reachable cardinal of the solution part.

The association of the two spaces, problem and solution, is reflected by the sum of these two cardinalities. Thus, the two values “Vc” and “Vr” will be defined as follows:

$$Vc = Vcp + Vcs$$
$$Vr = Vrp + Vrs$$

Furthermore, we associated the average coverage of the case base reflecting the average coverage rate of the entire base. This rate will serve as a mark for the introduction of a new case into the base. Average coverage is a good reference mark for learning cases in the case base because it characterizes the base's capacity and the problem resolution space. The goal is to increase this space so that the case base can cover more problems. In contrast, this increase may seriously damage the base's structural installation, hence the difficulty of dynamic learning on the part of cases. Consequently, the two steps of case-base maintenance and its auto-increment are strongly connected. The average coverage of the case base is given by the following formula:

$$Vc_{CB} = \frac{\sum_{i=0}^{n} Vc_i}{n}$$

Where "n" is the number of cases in the case base.

This average rate makes it possible to choose those cases which make the best possible contribution to the competence of the case base. This rate evolves with the evolution of the number of cases stored in the base.

3.2.2 Auto-increment.

The auto-increment stage is supported by an algorithm that takes into account the coverage and reachability values of both problem and solution parts and also serves as the VcCB baseline measure.

Auto-increment algorithm

When a case is to be added to the case base, a "targett*" is first created, comprised of a target T associated with its potential solution.

target=(Pt, ?) targett*=(Pt, St)

Once created, the "targett*" is ready to be learned. To introduce a "targett*" into the case base, its solution must not yet exist within the case base.

Let the problem part (symptom) be reachable by a number of cases lower than the mean coverage rate of the case base. This condition will ensure that the introduced case will contribute to the competence of the case base because its reachability rate will be relatively low.

The incremental learning algorithm is as follows:

1. **Let** the "CB" Case Base
2. target* case ← target case // change of the target case status by associating a solution
3. **For** each target* case do
   4. **If** Vrs > 0 **then** // the solution of the solved case is reachable by other solutions of the source cases
      5. **If** Vrp < Vc_{CB} then // the problem part which is reachable by the problem part of the source cases is higher than the average coverage rate of the CB
         6. Source Case ← targett* case
         7. CB ← CB $\cup$ source case

      EndIf

   ElseIf

   8. Source case ← target*case //Change of target* case status
   9. CB ← CB $\cup$ target* case // Introduce target* case in the CB

   EndIf

EndFor

Algorithm 4. Auto-increment algorithm of the case base

The operating principle of the algorithm is as follows:

According to the solution part reachability rate of the targett* case, this case will either be admitted to the case base or not. If "Vrs" is equal to zero, it means that no case similar to this newly solved one was previously listed in the base and it will thus be added. On the other hand, if the solution is reachable (Vrs>0) then we are interested in the problem part reachability rates compared to the average coverage rate of the case base. If the reachability rate is less than " Vc_{CB} " (which means that

the coverage of the solved case is higher than that of the case base) then the case is admitted for learning. This learned case will help to improve the coverage rate of the case base and thus to improve overall competence.

4. Assessment of the proposed method

4.1. Assessment of structuring method

In our study of case-base structuring we propose a finer categorization comprised of five (5) classes of case by splitting the category of the auxiliary cases into two subcategories: intra-class and inter-class.

In this section we compare the size, the reduction rate, performance and competence of the case-bases produced by means of different editing techniques used on a range of standard data-sets. The CNN algorithm was the first reduction technique for reference base size, based on static considerations (Dasarathy, 1991). The algorithm aims at reducing the entire input space into a representative subspace having the same properties.

In addition, this section is divided into two subcategories. The first is related to the competence study of the two forward selection strategies used in CNN and RC algorithms for comparison with the method proposed here. It should be noted that the RC method gives the best results in the case addition strategy. The second subcategory relates to the performance study of the proposed method. The ICF method produces better results than the others in backward elimination strategy.

4.1.1. Competence Study

Three different editing techniques are compared in this experimental study: 1) the standard CNN approach, 2) RC-CNN with cases placed in order according to their relative coverage values, and 3) CM with cases in order according to their CM values and associated algorithm. To reinforce the comparison, four different data sets are used. Travel (351 cases, 34 attributes) and Property (506 cases, 32 attributes) are traditional CBR data sets. The other two, Credit (690 instances, 15 attributes and 2 classes) and Ionosphere (351 instances, 34 attributes and 2 classes) represent classification problems. Property, Credit and Ionosphere data-sets are available from the UCI Machine Learning Repository (www.ics.uci.edu/mlearn/MLReposit ory.html; Blake et al., 1998).

A Travel data set is also available from the AI-CBR Archive (www.aicbr.org). In this section, the sizes of the case base in relation to their competence over unseen target problems are compared. As in (Smyth and McKenna, 1999), each editing strategy is used to generate a case base for the four used data-sets. This time, however, 100 random test problems are removed from the training set before the case-base construction. The final size of the case base and their competence over the 100 test problems is noted. The table below (Table 2) illustrates the comparison of the three editing techniques using the four data-sets.

Table 2. *Comparison of editing strategies over the test data sets.*

Data set	Property/Method	CNN	RC	CM
Travel	Mean case-base size	184.28	197	145.74
	Competence(%)	89.25	88.72	90.84
Property	Mean case-base size	55.19	57.81	39.62
	Competence(%)	95.92	95.53	95.91
Credit	Mean case-base size	344.84	297.4	215.76
	Competence(%)	58.85	58.95	62.37
Ionosphere	Mean case-base size	61.93	46.39	43.87

	Competence(%)	85.78	84.44	86.92

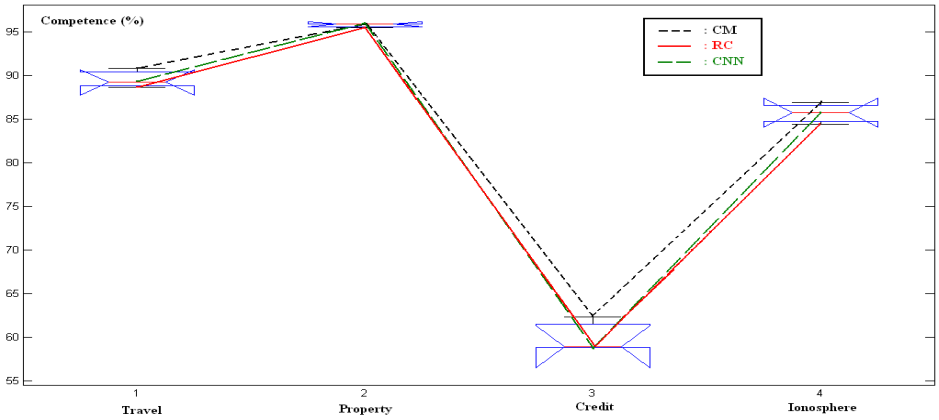

Fig. 5 (a) competence by one-way analysis of variance (ANOVAL)

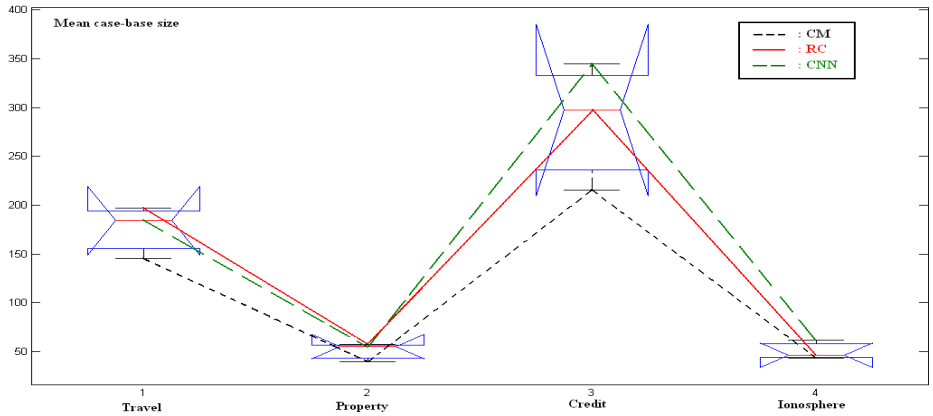

Fig. 5 (b)Mean case-base size by one-way analysis of variance (ANOVAL)

The results are positive. We can see it from Table 2 and from the figure 5 (a and b) defining the One-Way Analysis of Variance (ANOVAL)" and the box plot of the columns of X. It can be clearly seen that the CM method is more efficient than the others since it achieves a better case reduction rate with a finer competence for the four data-sets. The reduction rate obtained by the developed method is considerably higher than that produced by the four traditional methods, especially in classification problems represented by the "Credit" and "Ionosphere" data-sets.

Concerning competence value, this is higher than the corresponding case base obtained via other methods, though it is essentially the same as the traditional method for the "Property" data set. This shows that our method selects cases that are more competent than those selected by the other methods.

4.1.2. Performance Study

The performance study between the two methods (ICF and CM) was conducted on 18 data sets taken from the UCI repository of machine learning data bases (Brighton and Mellish, 2002b). Indeed, Brighton and Mellish gathered the compared methods into two groups in chronological order:

- Older methods: CNN, RNN, SNN, Chang, Wilson Editing, Repeated Wilson Editing, and All k –NN.

- Iterative case Filtering Algorithms: IB2, IB3, TIBLE, Cameron-Jones's Extensions and RT3 the most successful of Wilson and Martinez's algorithms.

The performance is evaluated based on the accuracy and the number of cases stored in the case base. According to Yan et al. (2014) and Dai et al. (2011), we used the five- fold cross validation. In our experiments, initially 20% of instances (randomly selected) are reserved for testing and the other 4* 20% =80% for training. This process is repeated five times. The average accuracy and mean size are given in Table 3. The accuracy and the resulting size are then calculated.

Table 3. Rate of performance and competence of the databases relating to CM as well as the comparison with IC

Data set	RT3		ICF		CM		Best method
	Storage (%)	Accuracy (%)	Storage (%)	Accuracy (%)	Storage (%)	Accuracy (%)	
anneal	20.72	91.82	22.59	91.35	20.05	**100.00**	CM
balance-scale	18.23	83.40	14.67	81.47	13.78	**95.83**	CM
breast-cancer-l	19.94	74.42	23.51	72.81	4.02	**96.56**	CM
breast-cancer-w	3.13	**95.26**	4.27	95.14	5.29	93.24	RT3ICF
cleveland	20.92	78.89	15.60	72.08	6.00	**91.01**	CM
credit	19.9	83.15	16.89	82.28	9.30	**88.76**	CM
glass	23.26	69.05	31.40	69.64	13.08	**72.51**	CM
hepatitis	19.15	83.33	16.33	82.26	11.03	**90.03**	CM
iris	16.04	**93.61**	42.08	92.56	10.66	86.99	RT3
lymphography	26.73	72.70	25.63	77.59	18.92	**96.31**	CM
mushrooms	5.50	**98.89**	12.80	98.64	14.65	98.22	RT3
Pima-indians	22.38	71.08	17.22	69.17	8.00	**93.09**	CM
post-operative	6.45	69.44	7.18	65.28	3.33	**83.46**	CM
thyroid	16.23	77.91	21.85	**86.63**	18.3	86.16	ICF
voting	7.43	93.77	8.88	91.19	2.50	**100.00**	CM
waveform	22.79	76.14	18.98	73.93	18.53	**96.87**	CM
wine	15.37	86.43	12.00	83.81	3.66	**92.94**	CM
zoo	26.13	87.08	52.78	92.42	18.81	**100.00**	CM
Average	17.24	82.58	20.25	82.13	11.66	**91.22**	CM

The results of filtering using ICF and RT3 are exactly those in which competence degrades as a result of noise removal. From the results in this table, several observations can be made. The CM method had very good storage reduction and generalization accuracy on average.

Some data sets seem to be especially well suited for the CM method. For example, it required less than 6% storage for the 6 data sets ("breast-cancer", "cleveland" "post-operative", "voting" and "wine"), yet it achieved a generalization accuracy that is even higher than the one of the ICF and RT3 methods. Generally, CM had higher average generalization accuracy than ICF and RT3, and also had the lowest storage requirements. However, "RT3" is slightly better than "CM" concerning accuracy and storage for two datasets: "breast-cancer-w" and "mushrooms". Regarding "ICF", it gets the best results for the "lymphography" datasets.

The final result (average storage and average accuracy) concerning the 18 data sets shows that the CM method is better than RT3 and ICF in terms of accuracy (91.22% against 82.13% and 82.58%) and its results for storage show that the CM method is only about half as large as the RT3 and ICF methods.

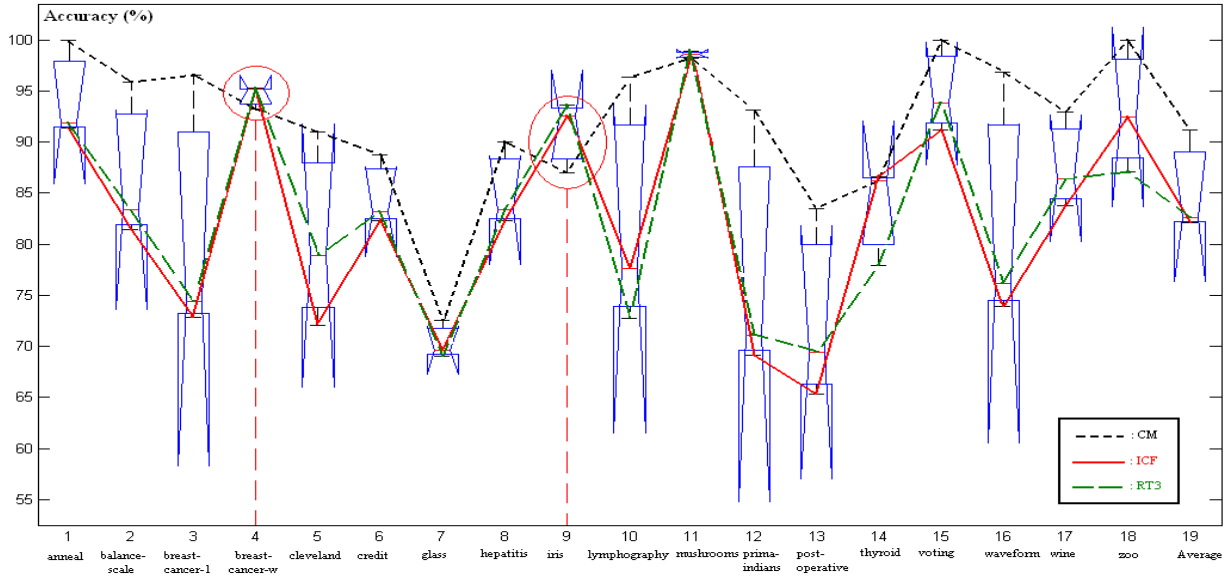

Fig. 6 (a)Accuracy by one-way analysis of variance (ANOVAL)

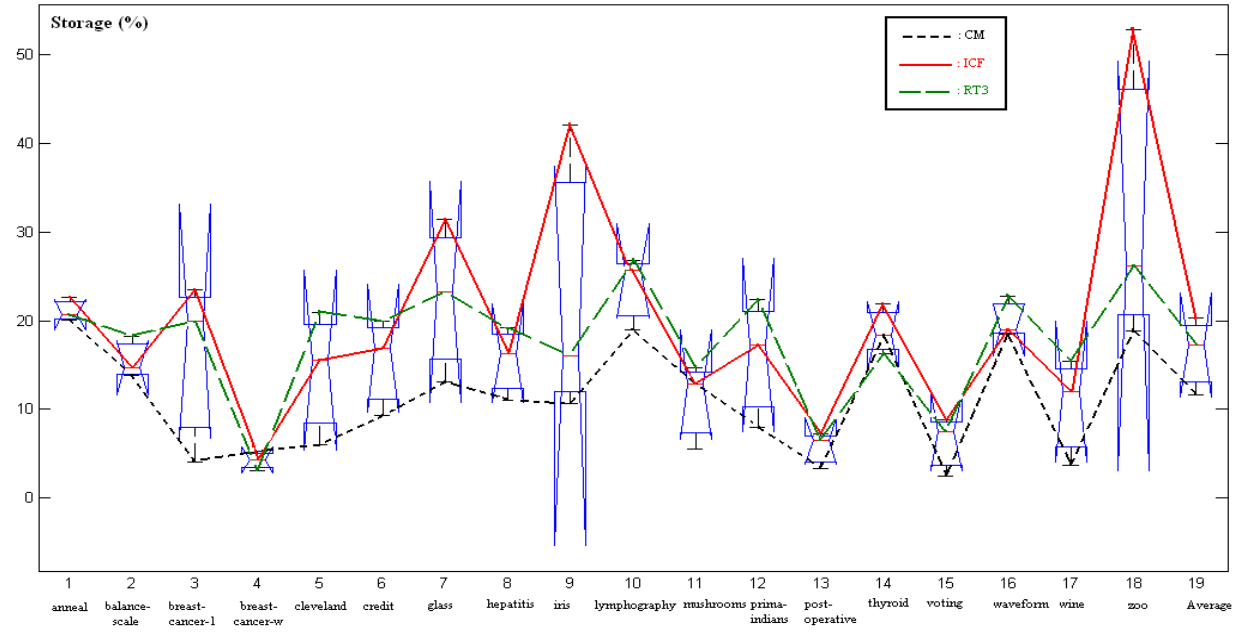

Fig. 6 (b)storage by one-way analysis of variance (ANOVAL)

Figures 6.a and b which are the "One-way analysis of variance"(ANOVAL) respectively, confirm that the CM method is the best. (It presents globally the best results in terms of the accuracy and case base size reduction as well.) However, we notice that the methods ICF and RT3 gives the best results for the iris and breast cancer benchmarks.

4.1.3. General study

We further evaluated the approach by testing the Eva (Ferrandiz and Boullé, 2010), and the RNNR-L1 (Dai and Hsu, 2011) algorithms. The latter belongs to the condensing type where the research strategy is decremental. According to its authors it enhances the results obtained by hybrid algorithms such as ICF, DROP3.

The Eva algorithm (Ferrandiz and Boullé, 2010) used with the NN algorithm which belongs to the clustering methods category presents promising results according to its author. The evaluation is done using five-fold cross validation as done in Dai's studies (Dai and Hsu, 2011). Ferrandiz and Boullé (2010) on the other hand, used ten-fold cross validation. This might bias the results as mentioned in Lupiani's recent work (Lupiani et al., 2014). But this gives us an estimation of methods' performances with respect to each other, without having to develop them.

Table 4. Comparison of different algorithms

	RT3		ICF		CM		Eva		RNNR-L1	
Data set	Storage (%)	Accuracy (%)	Storage (%)	Accuracy (%)	Storage (%)	Accuracy (%)	Storage (%)	Accuracy (%)	Storage (%)	Accuracy (%)
breast-cancer-w	3,13	95,26	4,27	95,14	5,29	93,24	**0,7**	**97,1**	18,65	**96,63**
iris	16,04	93,61	42,08	92,56	10,66	86,99	**2,3**	**96**	28	**96**
Pima-indians	22,38	71,08	17,22	69,17	8	**93,09**	**0,5**	73,4	25,86	69,4
waveform	22,79	76,14	18,98	73,93	18,53	**96,87**	**0,3**	79,9	24,61	77,42
wine	15,37	86,43	12	83,81	3,66	**92,94**	**2,2**	88,9	25,07	96,62
Average	15,942	84,504	18,91	82,922	9,228	**92,626**	**1,2**	87,06	24,438	87,214

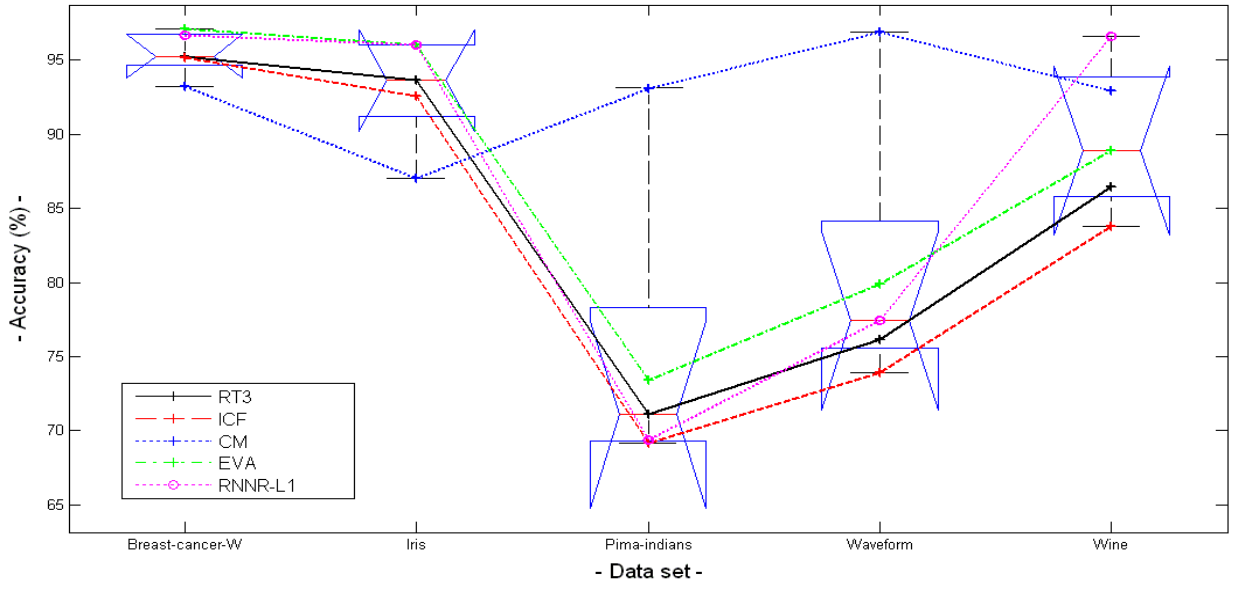

Fig. 7 (a) competence by one-way analysis of variance (ANOVAL)

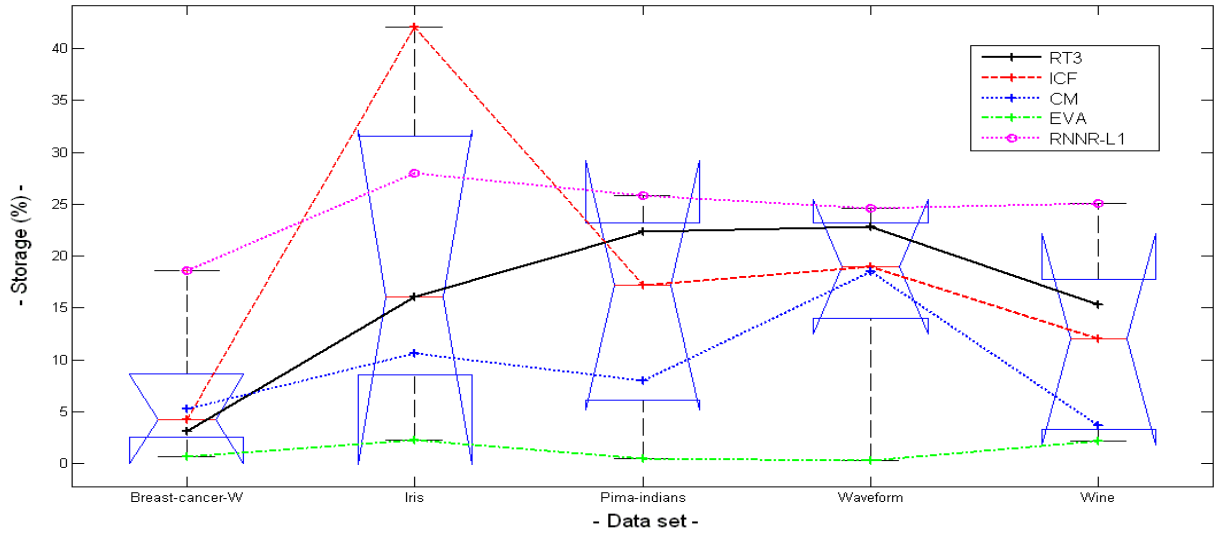

Fig. 7 (b) Mean case-base size by one-way analysis of variance (ANOVAL)

We can notice from the results defined by ANOVAL analysis that globally EVA and RNNR-L1 enhance the RT3 results in term of accuracy. CM on the other hand is distinguished by its better results for the Pima Waveform and Wine benchmarks. However, it gives less good results for the Iris and breast cancer datasets.

Concerning the storage, Eva presents the best results, it is followed by CM.

4.2. Assessment of the auto-increment method

The auto-increment method can be assessed by using the number of learned cases according to a defined protocol. The protocol is as follows: the case-base will be divided into two parts: the training-set contains 80% of the case base with the new case set containing the remaining 20% of cases and added to 10% of cases selected randomly from the training set. Consequently, the newly constituted set will contain 30% of cases from the case base (Fig. 8).

The cases contained in a test set are then subjected to the training set. If the case in the new case set meets the necessary conditions to be learned in the training base then it will be integrated by the latter, and so on for all cases of this set. Thereafter, we obtain the learned case base containing all cases from the training base as well as the cases from the test base having met the necessary conditions.

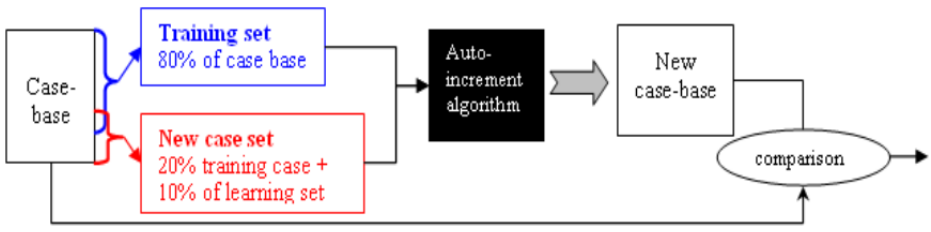

Fig. 8. Protocol set up concerning the evaluation of the auto-increment algorithm

Finally, the rate of the number of learned cases from the structured case-base will be calculated.

The learned cases are shown in Figure 9.

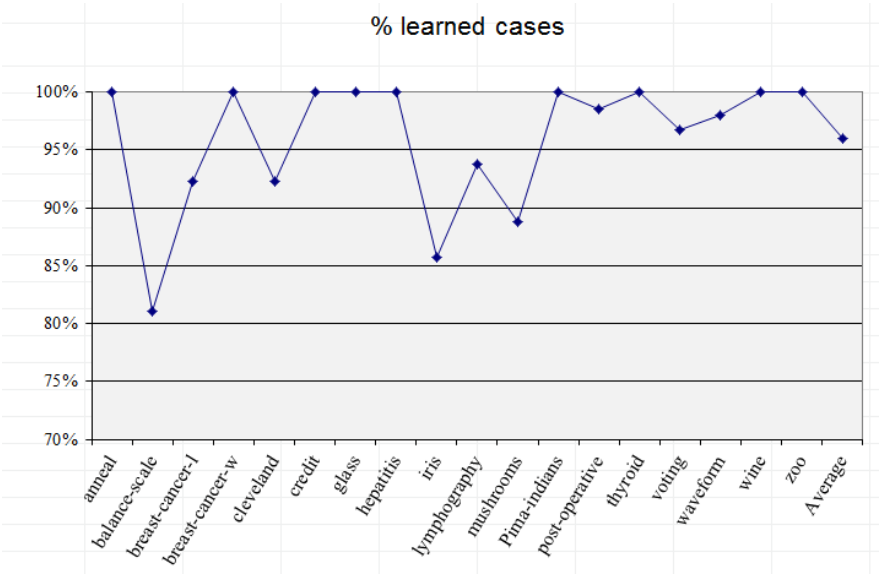

Fig. 9. Rate of learning of the various databases relating to the CM method.

It is noteworthy that the percentage of the learned cases (the number of learned cases resulting from the auto-increment out of the number of initial cases of the case base) varies from one database to another (from 81% to 100%). This means that one type of cases may not even be considered to be a database by another database. The data bases “balances-scale”, “iris” and “mushroom” are those whose learning rate is below 90%, which means that the auto-increment algorithm has detected cases that were not necessary to the case base. This is because the case-base maintenance was undertaken on cases completely independent of the base. As for the data bases with 100% learning ("anneal ", "credit", "glass" ...), this means that the auto-increment algorithm has performed perfectly by automatically learning all cases from the test base.

5. Application of the proposed method

The proposed method was applied to an industrial system, a supervised industrial system of pallet transfer (SISTRE) and was evaluated on a case base that relates the supervised industrial system of pallet transfer (SISTRE) (Haouchine et al., 2008).The case base maintenance was developed through minor adjustments to simple applications of the “data mining” type. This application is modeled by a case base containing attributes that are all informed. The values of these attributes constitute a hierarchy, which is operated during the adaptation phase.

SISTRE is composed of five robotized work stations which are served by a pallet transfer system organized into double rings (internal and external). Each station is equipped with pneumatic actuators (pushers, pullers and indexers) and electric actuators (stoppers) as well as a certain number of inductive sensors (proximity sensors). An inductive read/write module locates and identifies each pallet and provides information relative to required operation at a concrete station. Pallet conveyance is ensured by friction on belts activated by electric motors. Each pallet has a magnetic label that is used as embedded memory that can be read at each work station by means of magnetic read/write modules (Balogh) and that memorizes the product assembly sequence. These labels thus determine the pallet path through the system.

Pallets are conveyed on the interior ring which allows transit between the various stations. When the pallet is to be handled by a robot at the work station (information read on the label of the pallet), the latter is detoured onto the external ring where the work station is located.

The station is situated on the external ring and contains pneumatic and electric actuators as well as inductive sensors.

The SISTRE case base is composed of 750 cases, 12 attributes and 9 classes. In this case base, the class to be found is an equipment class to be repaired which is formalized in instance form. In the SISTRE base, the space is taken from the target cases as the total case base space.

The case descriptors are provided by components of different sources such as sensors and control and command units. A case is composed of eleven modal problem descriptors (ds: source descriptor problem) and one solution descriptor (Ds: source descriptor solution) reflecting a given failure class.

ds1: station ϵ {station1; station2; station3; station4; station5}.

ds2: zone 2 {internalring; externalring; postzone; pusherzone; pullerzone; robotzone}.
ds3: subzone { entry; internalconveyer; externalconveyer; robot; exit}.
ds4: component-equipment 2{ internalcarpet; externalcarpet; indexer; robot; pusher; puller}.
ds5: presence palette 2 {yes; no}.
ds6: sensor type _ {balogh0(bal0); balogh1(bal1); sensorDi; i = 19}.
ds7: sensor state 2 [0; 1].
ds8: stopper _ {Si; i = 16}.
ds9: stopper state 2 [0; 1].
ds10: context variable _ {balogh0(bal0); balogh1(bal1); sensorDi; i = 19}.
ds11: context variable state 2 [0; 1].
Ds1 : failure class 2 {sensor; stopper; pusher; puller; indexer; internalcarpet; externalcarpet; balogh; robot}.

	Problem											Solution
Ind	ds1	ds2	ds3	ds4	ds5	ds6	ds7	ds8	ds9	ds10	ds11	Ds1
2s	station 1	external ring	entry	ext carpet	present	D2	1	S2	0	D3	0	Stopper
9d	station 4	post zone	entry	Tapis ext	present	D5	0	S3	1	D4	0	Sensor
1t	station 5	pusher zone	entry	pusher	present	D6	1	S5	0	Ball	1	Pusher

Fig.10. *Example of three cases of the SISTRE case base.* (Haouchine et al., 2008).

The first case reflects a failure in the S2 stopper. The first four descriptors (ds1…ds4) determine the failure place which depends on the zone where the pallet is blocked. In fact, the failure is situated in the external carpet equipment which is at the entrance of the secondary ring of station 1. Then, the following eight descriptors (ds5…ds11) provide the component states involved in this area and their values. Finally, in the solution part, the "Ds1" descriptor specifies the failure class which is "stopper".

By applying the structuring algorithm, prior to the case deletion, the following results are obtained:

Table 5. Statistics of SISTRE case base

Type of case	Pivotal cases	Auxiliary cases	Inter-class spanning cases	Intra-class spanning cases	Support cases	Support Group
Number of cases in CB	80	120	50	155	345	86
Number of deleted cases	0	120	41	155	259	0
Number of kept cases	80	0	9	0	86	86

The maintenance method carries out the removal of the 120 auxiliary cases and the 155 intra-class spanning cases (Table 5). Concerning the support cases, our method keeps the 86 support cases representing each support group (each representative has the largest recovery space). It then removes the 41 inter-class spanning cases, keeping only 9 (corresponding to the number of failure classes). Finally, the method keeps the 80 pivotal cases. The results obtained appear in Table 5 and show a reduced case base containing 175 cases. A set of 140 cases will represent the training base and another set of 52 cases will represent the test base. By applying the principle of the incremental learning algorithm, we obtain the 52 cases to be learned, 35 of which were added to the training-base, with the remaining 17 considered similar. The results of structuring and the auto-increment of the case base are shown in Table 6.

Table 6. *Performance statistics, competence and auto-increment of the SISTRE case base.*

Initial case base size		750
Size of the obtained case base		175
case base Performance	Reduction ratio	76.67%
	Accuracy	100%
case base Competence		100%
Training-base size		140
Test-base size		52
Learning		100%

The competence rate is of 100% because the resulting case base solves the same number of problems as the initial one. The results are promising and, by applying the proposed method to the SISTRE case base, a reduced case base was obtained. Indeed, the case base is reduced by three-quarters (175 per 750 cases) while retaining the same initial competence. The results show the very good case-base performance which is in relation to the reduction ratio and the accuracy. This good performance is expressed through the decreasing retrieval time with a 100% accuracy. Concerning the learning, the 25 cases which were added to the training base, in which there were initially 140 cases, thus form a total of 175 cases. By comparing the cases of the resulting learned training base with the reduced case base, it has been found that there are exactly the same cases and the same number of cases. This gives a learning result of 100%. This result may appear excessive, but it corresponds to the test undertaken. Indeed, the cases that are not found in the training set (20%) will be added to it and the others are covered by the existing cases (10%). This explains why we find the same cases in the initial case base. Thus the auto-increment algorithm has been shown to have performed well by learning only the useful cases in the training case base. This algorithm led to the initial reduced case base by means of its automatic learning process. An optimal case base was therefore obtained.

6. CONCLUSION AND FUTURE WORK

Preserving the quality of the case base, both in terms of skill and performance, and ensuring its updating is a challenge that has been carried out using CBR-system maintenance tools. Indeed, case base maintenance (CBM) is crucial to ensuring the continuing success of a CBR system over time. In order to optimize and ensure a good quality case base, we have studied the already existing work in this field and have synthesized and compared their results concerning the following points: search direction, search procedure, selection criteria and stop criterion. This led us to propose a case base maintenance method drawing on the strong points of the various existing methods, namely the Backward Elimination method, the joint use of the two competence and performance criteria and the exploitation of a metric in support of Smyth and McKenna's categorization. We thus propose a case

base maintenance method incorporating two main stages, one offline stage enabling its structuring and one online stage enabling its auto-increment. The method developed offline is based on three steps: 1) ~~one of~~ case base evaluation through quality criteria, 2) one finer categorization of the competence model of Smyth and McKenna, relying on a competence measure and, in some cases, on case base performance, and 3) one of prototyping enabling the selection of the most relevant cases (pivotal cases, support cases according to their coverage and inter-auxiliary cases). This structuring aims at reducing the size of the case base by preserving its qualities: competence and performance. However, the CBR system works in incomplete environments in which they evolve through the emergence of new knowledge.

Therefore, from a partial case base, an updating mechanism has been developed which utilizes an auto-increment algorithm. This algorithm was used to insert new cases into the case base while respecting its structuring and quality. Indeed, the two proposed mechanisms for case based structuring and its auto-increment are complementary and contribute to the improvement of the CBR system and its evolution. The case base maintenance method is validated from several benchmarks and using the best methods.

The competence criterion was studied via four benchmarks and two methods, namely RC - CNN and CNN.

A study of the performance criterion was conducted on eighteen benchmarks and the ICF reference method. Finally, auto-increment was performed on the same eighteen previous benchmarks. Thus, the introduced method compares favorably to the most successful ones in existence and the results obtained were positive in terms of case base size reduction, accuracy and best competence. We applied our work to ~~two~~ industrial diagnosis case bases.

The first relates to a Supervised Industrial System of pallet Transfer (SISTRE) to which we applied the maintenance method.

In the future, we plan to undertake full-scale tests on CBR industrial systems in order to prove the feasibility of the proposed method. This will allow us to adapt our method to the continually changing needs of companies using CBR.

References

Aamodt, A., & Plaza, E. 1994. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications, 7(1), 39-59.

Aha, D.W., & Bankert, R., 1994. Feature selection for case-based classification of cloud types: an empirical comparison. In Aha, D.W., ed.: Proceedings of the AAAI-94 Workshop on Case-Based Reasoning, Menlo Park, CA: AAAI Press 106–112.

Arshadi, N., & Jurisica, I., 2004. Maintaining case-based reasoning systems: a machine learning approach. In Advances in Case-Based Reasoning (pp. 17-31). Springer Berlin Heidelberg.

Blake, C., Keogh, E., & Merz, C. J., 1998. UCI Repository of machine learning databases, Irvine: Department of Information and Computer Science, University of California.

Blum, A. L., & Langley, P., 1997. Selection of relevant features and examples in machine learning. Artificial intelligence, 97(1), 245-271.

Brighton, H., & Mellish, C., 2002a. On the consistency of information filters for lazy learning algorithms, in Principles of Data Mining and Knowledge Discovery, Third European Conference, PKDD '02, Proceedings, Prague, Czech Republic, September 15-18, 2002.

Brighton, H., & Mellish, C., 2002b. Advances in instance selection for instance-based learning algorithms. Data mining and knowledge discovery, 6(2), 153-172.

Bonzano, A., Cunningham, P., & Smyth, B., 1997. Using introspective learning to improve retrieval in CBR: A case study in air traffic control. In Case-Based Reasoning Research and Development (pp. 291-302). Springer Berlin Heidelberg.

Cao, G., Shiu, S. C., & Wang, X., 2001. A fuzzy-rough approach for case base maintenance, in: International Conference on Case Based Reasoning, pp. 118–130.

Calvo-Zaragoza, J., Valero-Mas, J. J., & Rico-Juan, J. R., 2014. Improving kNN multi-label classification in Prototype Selection scenarios using class proposals. Pattern Recognition.

Craw, S., Jarmulak, J., & Rowe, R., 2001. Maintaining Retrieval Knowledge in a Case□Based Reasoning System. Computational Intelligence, 17(2), 346-363.

Dasarathy, B. V., 1991. Nearest neighbor ({NN}) norms: {NN} pattern classification techniques. Los Alamitos, California. IEEE Press.

Dai, B. R., & Hsu, S. M. (2011). An instance selection algorithm based on reverse nearest neighbor. In Advances in Knowledge Discovery and Data Mining(pp. 1-12). Springer Berlin Heidelberg.

Duda, R. O., Hart, P. E., & Stork, D. G., 2001. Pattern Classification, Jon Wiley & Sons Inc. New York, 630-633.

Fazzolari, M., Giglio, B., Alcalá, R., Marcelloni, F., & Herrera, F., 2013. A study on the application of instance selection techniques in genetic fuzzy rule-based classification systems: Accuracy-complexity trade-off. Knowledge-Based Systems, 54, 32-41.

Ferrandiz, S., & Boullé, M. (2010). Bayesian instance selection for the nearest neighbor rule. Machine learning, 81(3), 229-256.

Garcia, S., Derrac, J., Cano, J. R., & Herrera, F., 2012. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 34(3), 417-435.

Hamidzadeh, J., Monsefi, R., & Yazdi, H. S. (2014). IRAHC: Instance Reduction Algorithm using Hyperrectangle Clustering. Pattern Recognition.

Haouchine, M. K., Chebel-Morello, B., & Zerhouni, N., 2008. Competence-Preserving Case-Deletion Strategy for Case-Base Maintenance. In ECCBR'08(Vol. 1, pp. 171-184).

Haouchine, M. K., 2009. Remémoration guidée par l'adaptation et maintenance des systèmes de diagnostic industriel par l'approche du raisonnement à partir de cas (Doctoral dissertation, Université de Franche-Comté).

Heister, F., & Wilke, W., 1998. An architecture for maintaining case-based reasoning systems. In Advances in Case-Based Reasoning (pp. 221-232). Springer Berlin Heidelberg.

Iglezakis, I., & Roth-Berghofer, T., 2000. A survey regarding the central role of the case base for maintenance in case-based reasoning. In ECAI Workshop Notes (pp. 22-28).

Leake, D. B., & Wilson, D. C., 1998. Categorizing case-base maintenance: Dimensions and directions. In Advances in Case-Based Reasoning (pp. 196-207). Springer Berlin Heidelberg.

Leake, D. B., & Wilson, D. C., 2000a. Guiding case-base maintenance: Competence and performance. In Proceedings of the 14th European Conference on Artificial Intelligence Workshop on Flexible Strategies for Maintaining Knowledge Containers.

Leake, D. B., & Wilson, D. C., 2000b. Remembering why to remember: Performance-guided case-base maintenance. In Advances in Case-Based Reasoning (pp. 161-172). Springer Berlin Heidelberg.

Liu, H., & Motoda, H., 2002. On issues of instance selection. Data Mining and Knowledge Discovery, 6(2), 115-130.

Markovitch, S., & Scott, P. D., 1988. The role of forgetting in learning. InMachine Learning: Proc. of the Fifth Intl. Conf (pp. 459-465).

Minton, S., 1990. Quantitative results concerning the utility of explanation-based learning. Artificial Intelligence, 42(2), 363-391.

Pękalska, E., Duin, R. P., & Paclík, P., 2006. Prototype selection for dissimilarity-based classifiers. Pattern Recognition, 39(2), 189-208.

Perner, P., 2006. Case-base maintenance by conceptual clustering of graphs.Engineering Applications of Artificial Intelligence, 19(4), 381-393.

Racine, K., & Yang, Q., 1996. On the consistency management of large case bases: the case for validation. In To appear in AAAI Technical Report-Verification and Validation Workshop (p.1).

Reinartz, T., 2002. A unifying view on instance selection. Data Mining and Knowledge Discovery, 6(2), 191-210.

Reinartz, T., Iglezakis, I., & Roth–Berghofer, T., 2001. Review and Restore for Case□Base Maintenance. Computational Intelligence, 17(2), 214-234.

M.M. Richter, 1998. Introduction, In Case-Based Reasoning Technology: From Foundations to Applications. Springer-Verlag, Berlin.

Smyth, B., & Keane, M. T., 1995. Remembering to forget. In Proceedings of the 14th international joint conference on Artificial intelligence (pp. 377-382).

Smyth, B., & Cunningham, P., 1996. The utility problem analysed (pp. 392-399). Springer Berlin Heidelberg.

Smyth, B., 1998. Case-base maintenance. In Tasks and Methods in Applied Artificial Intelligence (pp. 507-516). 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA-98-AIE. Proceedings. Springer Berlin Heidelberg.

Smyth, B., & McKenna, E., 1998. Modelling the competence of case-bases. InAdvances in Case-Based Reasoning (pp. 208-220). Springer Berlin Heidelberg.

Smyth, B., & McKenna, E. (1999). Building compact competent case-bases. InCase-based reasoning research and development (pp. 329-342). Springer Berlin Heidelberg.

Smyth, B., & McKenna, E., 2001a. Competence guided incremental footprint-based retrieval. Knowledge-Based Systems, 14(3), 155-161.

Smyth, B., & McKenna, E., 2001b. Competence models and the maintenance problem. Computational Intelligence, 17(2), 235-249.

Leyva, E., González, A., & Pérez, R., 2015. Three new instance selection methods based on local sets: A comparative study with several approaches from a bi-objective perspective. Pattern Recognition, 48(4), 1523-1537.

Lupiani, E., Juarez, J. M., & Palma, J. (2014). Evaluating Case-Base Maintenance algorithms. Knowledge-Based Systems, 67, 180-194.

M. Mykola Galushka, D. Patterson, 2006, Intelligent index selection for case-based reasoning, Knowledge-Based Systems 19 (8) 625–638.

Lu, N., Lu, J., & Zhang, G., 2009. An Integrated Knowledge Adaption Framework for Case-Based Reasoning Systems. In Knowledge-Based and Intelligent Information and Engineering Systems (pp. 372-379). Springer Berlin Heidelberg.

Roth-Berghfer, T. & Iglezzakis, I. (2001). Six Steps in Case-Based Reasoning: Towards a maintenance methodology for case-based reasoning systems. In Proceedings of the 9th German Workshop on CBR, GWCBR'01, Germany.

Salamó, M., & Golobardes, E., 2003. Hybrid Deletion Policies for Case Base Maintenance. In FLAIRS Conference (pp. 150-154).

Salamó, M., & Golobardes, E., 2004. Dynamic case base maintenance for a case-based reasoning system. In Advances in Artificial Intelligence–IBERAMIA 2004 (pp. 93-103). Springer Berlin Heidelberg.

Salamó, M., & López-Sánchez, M., 2011a. Rough set based approaches to feature selection for case-based reasoning classifiers. Pattern Recognition Letters, 32(2), 280-292.

Salamó, M., & López-Sánchez, M., 2011b. Adaptive case-based reasoning using retention and forgetting strategies. Knowledge-Based Systems, 24(2), 230-247.

Shiu, S. C., Yeung, D. S., Sun, C. H., & Wang, X. Z., 2001. Transferring Case Knowledge To Adaptation Knowledge: An Approach for Case□Base Maintenance. Computational Intelligence, 17(2), 295-314.

Smiti, A., & Elouedi, Z. (2014). WCOID-DG: An approach for case base maintenance based on weighting, clustering, outliers, internal detection and dbsan-gmeans. Journal of Computer and System Sciences, 80(1), 27-38.

Verbiest, N., Cornelis, C., & Herrera, F. (2013). FRPS: a fuzzy rough prototype selection method. Pattern Recognition, 46(10), 2770-2782.

Yang, Q., & Wu, J., 2000. Keep it simple: A case-base maintenance policy based on clustering and information theory. In Advances in Artificial Intelligence (pp. 102-114). Springer Berlin Heidelberg.

Yang, Q., & Zhu, J., 2001. A Case□Addition Policy for Case□Base Maintenance. Computational Intelligence, 17(2), 250-262.

Zhang, Z., & Yang, Q., 2001. Feature weight maintenance in case bases using introspective learning. Journal of Intelligent Information Systems, 16(2), 95-116.

Zhang, Z., & Yang, Q., 1999. Dynamic refinement of feature weights using quantitative introspective learning. In IJCAI (pp. 228-233).

Zhu, G. N., Hu, J., Qi, J., Ma, J., & Peng, Y. H., 2015. An integrated feature selection and cluster analysis techniques for case-based reasoning.Engineering Applications of Artificial Intelligence, 39, 14-22.